Desenvolvimento de Processos Workflow utilizando o Expresso Livre / Galaxia

Maurício Luiz viani

Sumário

- Interface de Monitoramento
- Metodologia de Desenvolvimento
- Arquitetura
- Implantação de Processos de Exemplo

Workflow

Interface de Monitoramento

Workflow - Interface de Monitoramento

Controle das instâncias dos processos

- Para cada processo o acesso é permitido individualmente
- Gráficos estatísticos

Workflow

Implantação dos Processos de Exemplo

Workflow - Implantação

Propósito

Este tipo de implantação não utiliza o ambiente de desenvolvimento. Tem como finalidade a simples disponibilização do processo em um novo ambiente.

Procedimento de Implantação

- Importar o XML do processo desejado
- Mapear os Perfis
- Ativar o Processo

Workflow

Metodologia de Desenvolvimento

Workflow - Metodologia de Desenvolvimento

- Pouca documentação (documentação mínima)
- Foco na organização do código

Workflow

Documentação Mínima de Projeto

Workflow - Documentação Mínima

Recomendação

Recomenda-se a utilização do ambiente wftrac.celepar.parana

Considerações

- O projetista também é o implementador
- Projetos possuem complexidade baixa ou média

Modelos e Documentos

- Levantamento de requisitos
- Diagrama de casos de uso
- Descrição dos casos de uso
- Fluxo do processo
- Diagrama entidade-relacionamento

Workflow

Padrões de Codificação PHP, Comentários e Banco de Dados

Workflow - Padrões de Codificação: PHP

Tags PHP

- Sempre utilizar tags do tipo `<?php ?>`

Indentação e Comprimento de Linha

- Tentar manter linhas com tamanho máximo de 80 caracteres
- Utilizar tabulação para indentação

Comentário

- Utilizar comentários de documentação (estilo PHPDocumentor)
- Quando necessário, comentar trechos do código que se apresentem mais complexos
- Utilizar comentários no padrão C/C++

Incluindo Código

- Usar require_once para inclusão incondicional
- Usar include_once para inclusão condicional

Workflow - Padrões de Codificação: PHP

Classes

- Devem possuir nomes descritivos. Cada palavra é iniciada em letra maiúscula. e.g., `Solicitacao, MinhaClasse`

Métodos e Funções

- Devem possuir nomes que indicam ação. Com exceção da primeira palavra, as outras devem ser iniciadas com letra maiúsculas. e.g., `gerarRelatorio(), finalizar()`

Constantes

- Devem ser escritas em letras maiúsculas e utilizar *underscore* como separador de palavras. e.g. `MINHA_CONSTANTE`
- Podem ser prefixadas com o nome de pacotes/classe. e.g. `VALIDACAO_EMAIL`

Workflow - Padrões de Codificação: PHP

Estruturas de Controle: if, for, while, switch, foreach, etc.

- Devem sempre possuir espaço entre a palavra-chave e a abertura de parênteses
- A abertura de chaves deve ser feita na mesma linha da estrutura de controle
- Recomenda-se sempre a utilização de chaves, mesmo quando opcional

Exemplo:

```
if ((condition1) || (condition2)) {
    action1;

} elseif ((condition3) && (condition4)) {

    action2;

} else {
    defaultaction;

}
```

```
switch (condition) {
    case 1:
        action1;
        break;

    case 2:
        action2;
        break;

    default:
        defaultaction;
        break;
}
```

Workflow - Padrões de Codificação: PHP

Chamadas de Função

Só possuem espaço após a vírgula que separa os parâ-metros que estão sendo passados

Definição de Funções/Métodos e Classes

- A abertura das chaves fica na linha seguinte ao nome da Função/Método/Classe
- O fechamento das chaves fica na linha seguinte da última linha de código.

Workflow - Padrões de Codificação: PHP

Comentários:

Define

```
/**
* Prefixo das tabelas do workflow
* @name GALAXIA_TABLE_PREFIX
*/
define('GALAXIA_TABLE_PREFIX', 'egw_wf_');
```

Funções

```
/**
* Includes files from the process folder.
* @param string $file_name File's name to be included.
* @return void
* @license http://www.gnu.org/copyleft/gpl.html GPL
* @access public
*/
function wf_include( $file_name )
```

Workflow - Padrões de Codificação: PHP

Comentários:

Classes

```
/**
* Criptografia simples e segura baseada em funções hash
* @author   Carlos Eduardo Nogueira Gonçalves
* @version 1.0
* @license http://www.gnu.org/copyleft/gpl.html GPL
* @package Workflow
**/
class Pessoa
```

Workflow - Padrões de Codificação: PHP

Comentários:

Atributos

```
/**
* @var string $hash_key Versão embaralhada da
* chave de criptografia fornecida pelo usuário
* @access public
**/
var $hash_key;
```

Métodos

```
/**
* Usado para definir a chave de criptografia
* @param string $key Chave secreta usada para criptografia
* @param boolean $base64 Usar codificação base64
* @return void
* @access public
*/
function setKey($key, $base64 = true)
```

Workflow - Padrões de Codificação: Banco de Dados

Comandos SQL

- Utilizar letras maiúsculas para as palavras reservadas da linguagem
- Utilizar letras minúsculas para nomes particulares do processo
- Separe palavras e prefixos com *underscore*
- Evite utilizar números
- Indentar código SQL caso a consulta seja grande ou complexa

Exemplo

```
SELECT
    usuario, data_hora
FROM
    ligacao_telefonica
WHERE
    (telefone = '102') AND
    (area_id = 8)
```

Workflow - Padrões de Codificação: Banco de Dados

Tabela

- Utilizar nomes curtos e no singular
- Evitar utilização de acrônimos
- Prefixe tabelas de metadados com o nome das tabelas com as quais se relacionam

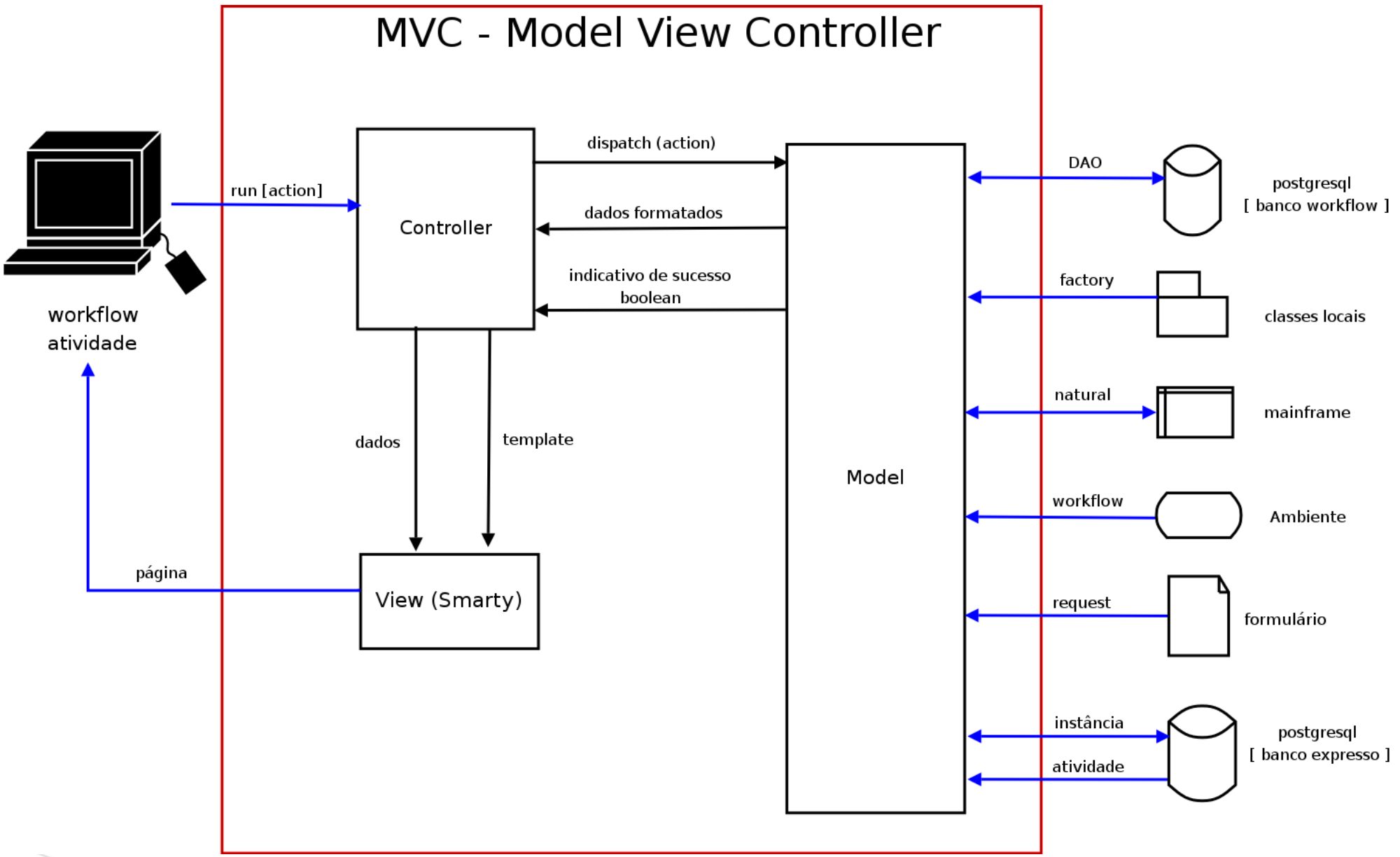
Campos e Colunas

- A chave primária deve possuir, quando possível, o nome da tabela com o sufixo “_id”.
- Em caso de chave estrangeira, deve-se manter o nome da chave primária da tabela referenciada
- Em caso de mais de uma ocorrência da mesma chave estrangeira em uma mesma tabela, prefixar as chaves estrangeiras com uma “descrição” apropriada

Workflow

Arquitetura

Workflow - Arquitetura: MVC (Model-View-Controller)



Workflow - Arquitetura: MVC (Model-View-Controller)

Módulo (BaseModel e BaseController)
Atua em todos os processos desenvolvidos no padrão MVC

Processo (Model e Controller)
Estende o nível anterior. Atua sobre as atividades do processo

Atividade (<Activity>Model e <Activity>Controller)
Estende os níveis anteriores. Recebe e responde as requisições dos usuários do processo.

Workflow - MVC: Camada View

Conceitos

- É toda a interface do sistema. Pode ser tanto destinada a pessoas quanto a outros programas.
- Manipulada diretamente apenas pela camada *Controller*.

Estrutura

- Representada por uma instância da *engine* de templates Smarty.

Workflow - MVC: Camada Controller

Conceitos

- Faz o gerenciamento entre o usuário, *Views* e *Model*.
- Deve saber quais são as funções do sistema e não como implementá-las.

Estrutura

Módulo

- Contém instâncias das outras camadas.
- Contém métodos que gerenciam ações padrões

Processo

- Possui atributos que indicam arquivos de *template* utilizados pelo processo

Atividade

- Gerencia a comunicação entre o usuário e o sistema.

Workflow - MVC: Camada Controller

Atributos

- templateFile
- view
- model

Métodos

- showForm()
- loadViewVars()
- dispatch()
- __default()
- run()

Workflow - MVC: Camada Model

Conceitos

- Define as regras de negócio do processo.
- Possibilita o desenvolvimento de recursos que independem da forma como os dados são exibidos.

Estrutura

Módulo

- Contém uma série de objetos importantes no desenvolvimento dos processos.

Processo

- Contém informações sob a forma de atributos

Atividade

- Contém métodos que implementam as ações solicitadas pelo usuário

Workflow - MVC: Camada Model

Atributos

- instance
- activity
- DAO
- request
- workflow

Métodos

- getAtributes()
- addViewVar()
- setWfProperty()
- getWfProperty()
- setNameInstance()
- updateInstance()
- updateAttributes()
- commitIntance()

Workflow - Arquitetura

Shared

- É incluído em todas as atividades do processo
- *Includes* devem ser feitos neste arquivo
- Definições de constantes também devem ser feitas neste arquivo.

Organização

- Apenas uma classe por arquivo .php
- Includes somente no arquivo shared.php

Workflow - Arquitetura: Nomenclatura

Atividade

Devem expressar uma ação. Primeira letra em maiúscula

Arquivos

Atividades: mesmo nome da atividade seguido de ".php"
Templates: mesmo nome da atividade seguido de ".tpl"
Classes: formato "class.nome.da.classe[.camada].inc.php"

Classes

Iniciadas com letra maiúscula
Subclasse de camada: NomeAtividade[Camada]

Métodos

Letras minúsculas mas, se composto por mais de uma palavra, as seguintes devem possuir a primeira letra maiúscula.
Métodos de subclasses do MVC:

Controller: [nomeAção] **Model**: [NomeAção]Action

Workflow - Arquitetura: Práticas

Strings pré-definidas

Valores, em string, pré-definidos, devem ser declarados em atributos internos escritos em maiúscula. Ou seja, devem atuar como constantes.

Ordem de Declaração em uma Classe

As declarações devem seguir a ordem:

1. Declaração de constantes
2. Declaração de atributos
3. Declaração de construtores
4. Declaração de métodos

Workflow

Ambiente de Desenvolvimento

Workflow - Ambiente de Desenvolvimento

Recomendações

Três Servidores:

- Desenvolvimento
- Homologação
- Produção

Desenvolvimento (Duas Partes):

Estação Desenvolvedor:

- Apache
- Php

Servidor Desenvolvimento:

- Apache
- Ldap
- Banco de dados
- Cyrus e Postfix

Workflow - Ambiente de Desenvolvimento

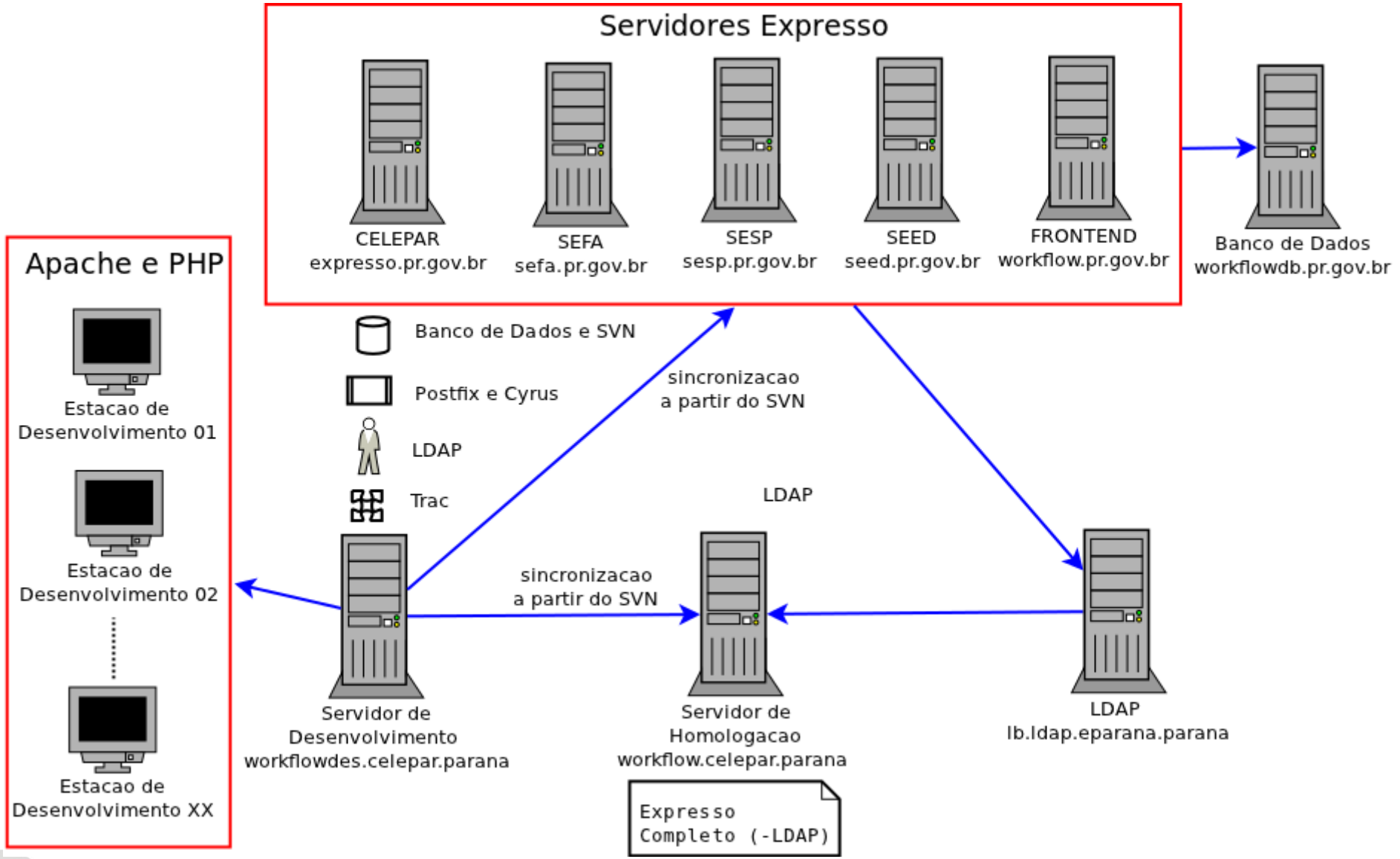
Homologação

- Apache
- Banco de dados
- Ldap e E-Mail compartilhados com produção

Produção

- Apache
- Banco de dados
- Ldap
- Cyrus e Postfix

Workflow - Ambiente de Desenvolvimento da Celepar



Workflow - Ambiente de Desenvolvimento do Laboratório

Desenvolvimento no laboratório

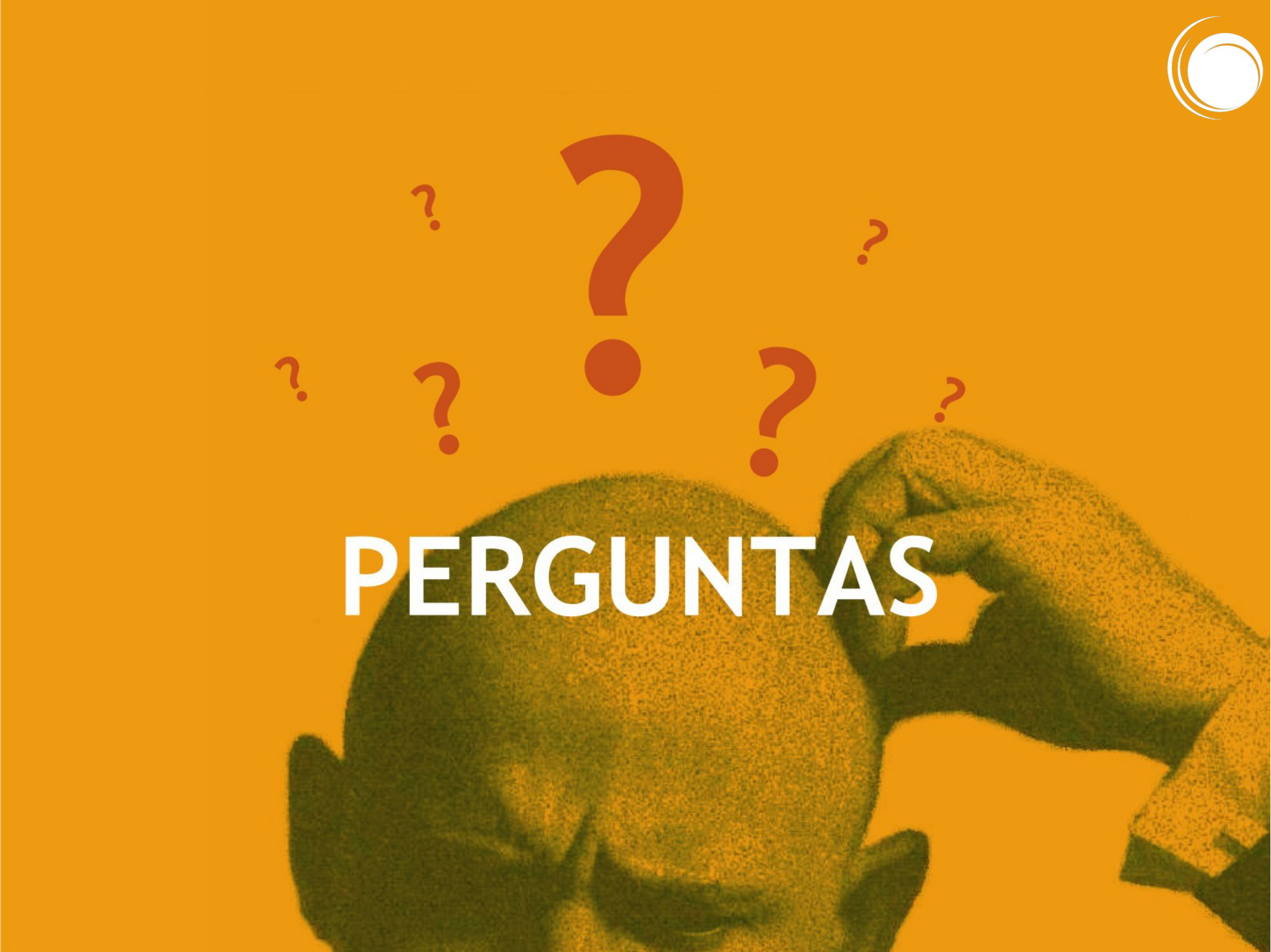
Estações de trabalho:

- SO: Linux
- Softwares: vim

Servidor de desenvolvimento

- Apache
- Ldap
- Banco de dados
- Cyrus
- Postfix

PERGUNTAS



Maurício Luiz Viani
viani@celepar.pr.gov.br
Projeto Workflow

Documentação do Workflow
http://trac.expressolivre.org/wiki/WF/Documentacao